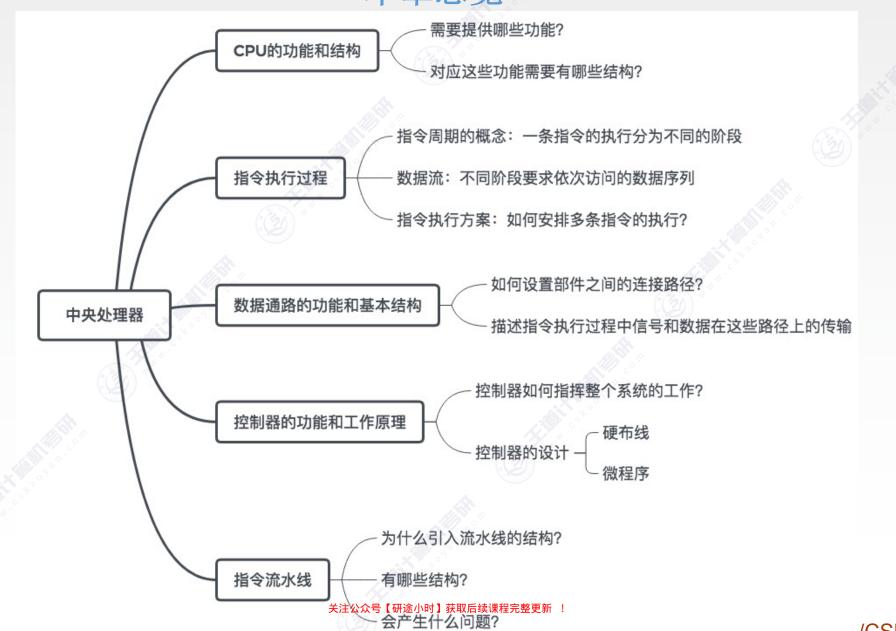
## 本节内容

控制器设计

硬布线控制器

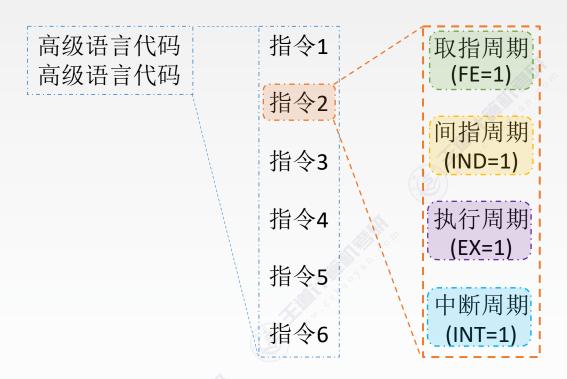
## 本章总览



根据指令操作码、目前的机器周期、节 拍信号、机器状态条件,即可确定现在 这个节拍下应该发出哪些"微命令"

### 内容回顾

CU发出一个微命令,可完成对应微操作。 如: 微命令1 使得 PC<sub>out</sub>、MAR<sub>in</sub> 有效。 完成对应的微操作1 (PC)→MAR



T<sub>0</sub>: 微操作1、微操作2

T<sub>1</sub>: 微操作3 T<sub>2</sub>: 微操作4 一个节拍内可以并行完成 多个"相容的"微操作

To: 微操作5、微操作2

T<sub>1</sub>: 微操作6 T<sub>2</sub>: 微操作7 同一个微操作可能在不同指令的不同阶段被使用

 ---- 节拍数各不相

 ---- 化设计,选择

| T<sub>1</sub>: 微操作8 | T<sub>2</sub>: 微操作9、微操作6 不同指令的执行周期所需 节拍数各不相同。为了简 化设计,选择定长的机器 周期,以可能出现的最大 节拍数为准(通常以访存 所需节拍数作为参考)

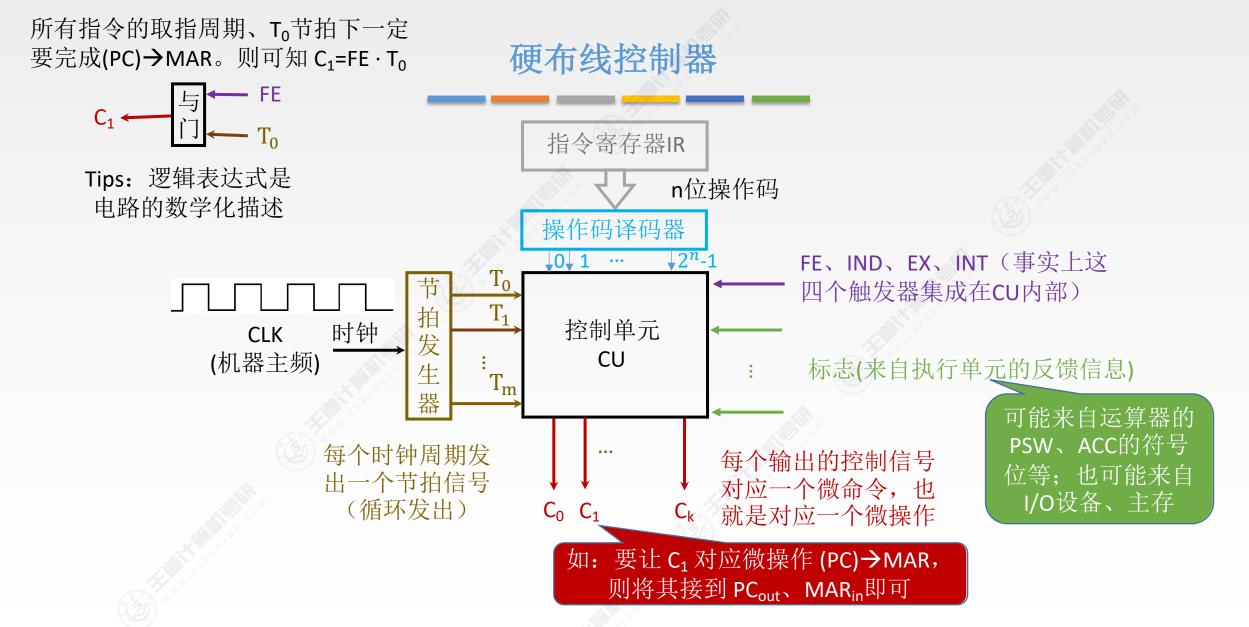
CLK  $1 \rightarrow FE$   $1 \rightarrow IND$   $1 \rightarrow EX$   $1 \rightarrow INT$ 

 $T_0$ :

 $T_0$ :

T<sub>1</sub>: 微操作10 T<sub>2</sub>: 微操作11 若实际所需节拍数较少, 可将微操作安排在机器周 期末尾几个节拍上进行

王道考研/CSKAOYAN.COM



根据指令操作码、目前的机器周期、节拍信号、机器状态条件,即可确定现在这个节拍下应该发出哪些"微命令"



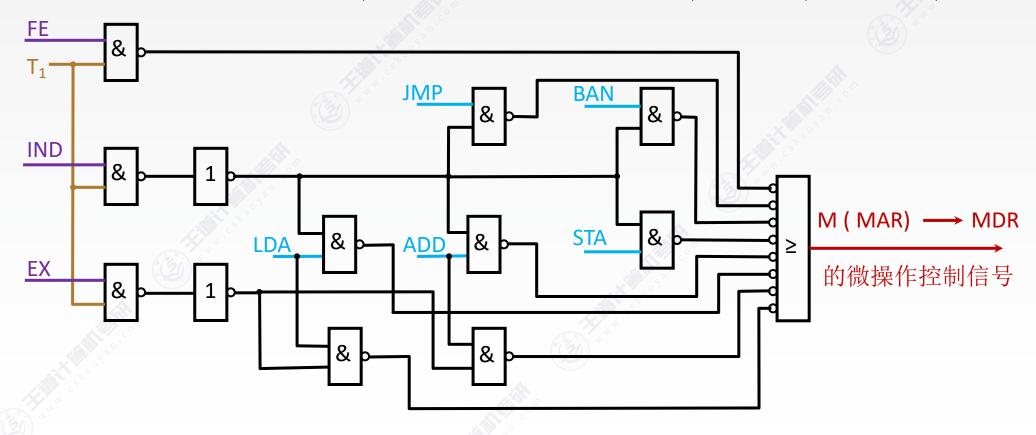
## 颤抖吧! 感受恐惧!



注:一般不考电路,莫慌~

M (MAR) →MDR 微操作命令的逻辑表达式:

 $FE \cdot T_1 + IND \cdot T_1(ADD + STA + LDA + JMP + BAN) + EX \cdot T_1(ADD + LDA)$ 



根据指令操作码、目前的机器周期、节拍信号、机器状态条件,即可确定现在这个节拍下应该发出哪些"微命令"

### 硬布线控制器的设计

#### 设计步骤:

确定哪些指令在什么阶段、在什么 条件下会使用到的微操作

- 1. 分析每个阶段的微操作序列(取值、间址、执行、中断四个阶段)
- 2. 选择CPU的控制方式

采用定长机器周期还是不定长机器周期?每个机器周期安排几个节拍?

3. 安排微操作时序

如何用3个节拍完成整个机器周期内的所有微操作?

4. 电路设计

假设采用同步控制方式(定长机器周期), 一个机器周期内安排3 个节拍。

安排,必须安排



注:中断周期内的微操作序列就不分析了,原理类似

## 分析每个阶段的微操作序列

取指周期(所有指令都一样)

 $PC \rightarrow MAR$ 

 $1 \rightarrow R$ 

 $M(MAR) \rightarrow MDR$ 

 $MDR \rightarrow IR$ 

OP (IR)  $\rightarrow$  ID

 $(PC) + 1 \rightarrow PC$ 

间址周期(所有指令都一样)

 $Ad(IR) \rightarrow MAR$ 

 $1 \rightarrow R$ 

 $M(MAR) \rightarrow MDR$ 

 $MDR \rightarrow Ad(IR)$ 

注: ID 是指令译码器 Instruction Decoder

罗列出<mark>所有</mark>指令在各个阶段的微操作序列,就可以知道 在什么情况下需要使用这个微操作

根据指令操作码、目前的机器周期、节拍信号、机器状态条件,即可确定现在这个节拍下应该发出哪些"微命令"

执行周期(各不相同)

注:很多地方把 ACC简写为AC

CLA

 $0 \rightarrow AC$ 

clear ACC 指令 ACC清零

LDA X

取数指令,

把X所指内容取到ACC

Ad (IR)  $\rightarrow$  MAR

 $1 \rightarrow R$ 

 $M(MAR) \rightarrow MDR$ 

 $MDR \rightarrow AC$ 

JMP X 无条件转移  $Ad(IR) \rightarrow PC$ 

负数符号位为1

BAN X

 $A_0 \bullet Ad (IR) + \overline{A_0} \bullet (PC) \rightarrow PC$ 

Branch ACC Negative 条件转移,当ACC为负时转移



## 安排微操作时序的原则

原则一 微操作的 先后顺序不得 随意 更改

原则二 被控对象不同的微操作

尽量安排在 一个节拍 内完成

原则三 占用 时间较短 的微操作

尽量 安排在 一个节拍 内完成

并允许有先后顺序

### 安排微操作时序-取指周期

微操作的 先后顺序不得 随意 更改 原则一

原则二 被控对象不同 的微操作

尽量安排在 一个节拍 内完成

占用 时间较短 的微操作 原则三

尽量 安排在 一个节拍 内完成

并允许有先后顺序

- (1) PC  $\rightarrow$  MAR
- (2) 1  $\rightarrow$  R
- 在(1)之后 (3) M (MAR)  $\rightarrow$  MDR
- (4) MDR  $\rightarrow$  IR
- (5) OP (IR)  $\rightarrow$  ID
- (6) ( PC ) + 1  $\rightarrow$  PC

存储器空闲即可

在(3)之后

在(4)之后

在(1)之后

### 安排微操作时序-取指周期

原则一 微操作的 先后顺序不得 随意 更改

原则二 被控对象不同的微操作

尽量安排在 一个节拍 内完成

原则三 占用时间较短的微操作

尽量 安排在一个节拍 内完成

并允许有先后顺序

 $T_0$  (1) PC  $\rightarrow$  MAR

 $T_0$  (2) 1  $\rightarrow$  R

T<sub>1</sub> (6) ( PC ) + 1 → PC 在(1)之后

T<sub>2</sub> (4) MDR → IR 在(3)之后

(5) OP (IR) → ID 在(4)之后

两个微操作占用时 间较短,根据原则 三安排在一个节拍

M(MAR)→MDR 从主存取数据,用时较长,因此必须一个时钟周期才能保证微操作的完成

MDR → IR 是CPU内部寄存器的数据传送,速度很快,因此在一个时钟周期内可以紧接着完成 OP (IR) → ID。也就是可以一次同时发出两个微命令。

存储器空闲即可

## 安排微操作时序-间址周期

原则一 微操作的 先后顺序不得 随意 更改

原则二 被控对象不同的微操作

尽量安排在 一个节拍 内完成

原则三 占用时间较短的微操作

尽量 安排在一个节拍 内完成

并允许有先后顺序

- $T_0$  (1) Ad(IR)  $\rightarrow$  MAR
- $T_0$  (2) 1  $\rightarrow$  R
- $T_1$  (3) M (MAR)  $\rightarrow$  MDR
- $T_2$  (4) MDR  $\rightarrow$  Ad(IR)

### 安排微操作时序-执行周期

原则一 微操作的 先后顺序不得 随意 更改

原则二 被控对象不同的微操作

尽量安排在 一个节拍 内完成

原则三 占用时间较短的微操作

尽量 安排在 一个节拍 内完成

并允许有先后顺序

```
① CLA
                     \mathsf{T}_0
clear
                            0 \rightarrow AC
ACC清零
2 COM
                     \mathsf{T}_0
complement T<sub>1</sub>
                            \overline{AC} \rightarrow AC
                     T_2
ACC取反
③ SHR
                     \mathsf{T}_0
                     \mathsf{T}_1
shift
                            L(AC) \rightarrow R(AC)
算术右移
                            AC_0 \rightarrow AC_0
4 CSL
                     \mathsf{T}_0
                     \mathsf{T}_1
cyclic shift
                           R(AC) \rightarrow L(AC), AC_0 \rightarrow AC_n
循环左移
⑤ STP
                     \mathsf{T}_0
                     \mathsf{T}_1
stop
                           0 \rightarrow G
停机
```

## 安排微操作时序-执行周期

### (1) 非访存指令

- ① CLA  $T_0$  clear  $T_1$   $T_2$   $0 \rightarrow$  AC
- $\bigcirc$  COM  $\mathsf{T}_0$  complement  $\mathsf{T}_1$
- ACC取反  $T_2$   $\overline{AC} \rightarrow AC$
- 3 SHR  $T_0$  Shift  $T_1$
- 算术右移  $T_2$  L(AC) → R(AC)
  - $T_2 AC_0 \rightarrow AC_0$
- 4 CSL  $T_0$  cyclic shift  $T_1$
- 循环左移  $T_2$  R(AC)  $\rightarrow$  L(AC), AC<sub>0</sub>  $\rightarrow$  AC<sub>n</sub>
- $\begin{array}{ccc} \boxed{5} \text{ STP} & T_0 \\ \text{stop} & T_1 \end{array}$
- 停机  $T_2 \rightarrow G$

### (2) 访存指令

- ⑥ ADD X  $T_0$  Ad (IR) → MAR, 1 → R
- 加法指令 T<sub>1</sub> M(MAR)→MDR
- 隐含ACC  $T_2$  (AC)+(MDR)  $\rightarrow$  AC
- $\bigcirc$  STA X  $_{10}$  Ad (IR) → MAR, 1 → W
- 存数指令 T<sub>1</sub> AC → MDR
- 隐含ACC  $T_2$  MDR → M (MAR)
- 8 LDA X  $T_0$  Ad (IR) → MAR, 1 → R
- 取数指令 T<sub>1</sub> M (MAR) → MDR
- 隐含ACC T<sub>2</sub> MDR → AC

#### (3) 转移指令

- ⑨ JMP X T<sub>0</sub> T<sub>1</sub> T<sub>1</sub> Ad (IR) → PC
- ① BAN X To Branch ACC To Negative To

条件转移

 $\frac{\mathsf{T}_1}{\mathsf{T}_2}$   $\mathsf{A}_0 \bullet \mathsf{Ad} (\mathsf{IR}) + \overline{\mathsf{A}_0} \bullet (\mathsf{PC}) \to \mathsf{PC}$ 

## 安排微操作时序-中断周期

原则一 微操作的 先后顺序不得 随意 更改

原则二 被控对象不同的微操作

尽量安排在 一个节拍 内完成

原则三 占用时间较短的微操作

尽量 安排在一个节拍 内完成

并允许有先后顺序

#### 设计步骤:

- 1. 分析每个阶段的微操作序列
- 2. 选择CPU的控制方式
- 3. 安排微操作时序
- 4. 电路设计

 $\sqrt{1}$  (1) a  $\rightarrow$  MAR

 $\Gamma_0$  (2) 1  $\rightarrow$  W 存储器空闲即可

T<sub>0</sub> (3) 0 → EINT 硬件关中断

T<sub>1</sub> (4) (PC) → MDR 内部数据通路空闲即可

T<sub>2</sub> (5) MDR → M(MAR) 在(3)之后

T<sub>2</sub> (6) 向量地址 → PC 在(3)之后

这些操作由中断隐指令完成

注:中断隐指令不是一条指令,而是指一条指令的中断周期由硬件完成的一系列操作

中断周期的三个任务:

- 1. 保存断点
- 2. 形成中断服务程序的入口地址
- 3. 关中断

## 组合逻辑设计

设计步骤:

1. 列出操作时间表

列出在取指、间址、执行、中断周期,T0、T1、T2 节拍内有可能用到的所有微操作

- 2. 写出微操作命令的最简表达式
- 3. 画出逻辑图

# 组合逻辑设计

设计步骤:

1. 列出操作时间表

非访存指令

工作 周期 标记	节拍	状态 条件	微操作命令信号	CLA	COM	SHR	CSL	STP	ADD	STA	LDA	JMP	BAN
	Т		PC → MAR	1	1	1	1	1	1	1	1	1	1
	$T_0$		$1 \longrightarrow R$	1	1	1	1	1	1	1	1	1	1
	T		$M(MAR) \rightarrow MDR$	1	1	1	1	1	1	1	1	1	1
FE	$T_1$		$(PC)+1 \longrightarrow PC$	1	1	1	1	1	1	1	1	1	1
取指	T <sub>2</sub>	o file	$MDR \rightarrow IR$	1	1	1	1	1	1	1	1	1	1
		1 1 1 2 2 7 CO.	$OP(IR) \rightarrow ID$	1	1	1(3)	1	1	1	1	1	1	1
		I	1→ IND		ji)	gh a			1	1	1	1	1
		/ T	$1 \longrightarrow EX$	1	1	1	1	1	1	1	1	1	1

# 组合逻辑设计

设计步骤:

1. 列出操作时间表

非访存指令

工作 周期 标记	节拍	状态 条件	微操作命令信号	CLA	СОМ	SHR	CSL	STP	ADD	STA	LDA	JMP	BAN
	$T_0$		$Ad(IR) \rightarrow MAR$						1	1	1	1	1
IND			$1 \longrightarrow R$					60	1	1	1	1	1
间址	$T_1$		$M(MAR) \rightarrow MDR$						1	1	1	1	1
	$T_2$		$MDR \longrightarrow Ad (IR)$					* 2 	1	1	1	1	1
		IND	$1 \longrightarrow EX$						1	1	1	1	1

间址周期标志

### 设计步骤:

# 组合逻辑设计

- 1. 列出操作时间表
- 2. 写出微 操作命令的 最简表达式

	工作 周期 标记	节拍	状态 条件	微操作命令信号	CLA	СОМ	ADD	STA	LDA	JMP	BAN
`				$Ad(IR) \rightarrow MAR$			1	1	1	A C	7
		$T_0$		$1 \longrightarrow R$			1		1		
				$1 \longrightarrow W$				1			
		J		$M(MAR) \rightarrow MDR$			1	A Clark	1		
	EX 执行	$T_1$	-16X	$AC \longrightarrow MDR$				1			
	17(1)	(0		(AC)+(MDR)→AC			1	off.			
			<i>2)</i>	$MDR \longrightarrow M(MAR)$				1			
	A	A CONTRACTOR		MDR→AC			. A		1		
		$T_2$		0→AC	1						
	A A A			$\overline{AC} \rightarrow AC$		° 1					
	9)			$Ad(IR) \rightarrow PC$	120404					1	
			$A_0$	$Ad(IR) \rightarrow PC$	. ·						1

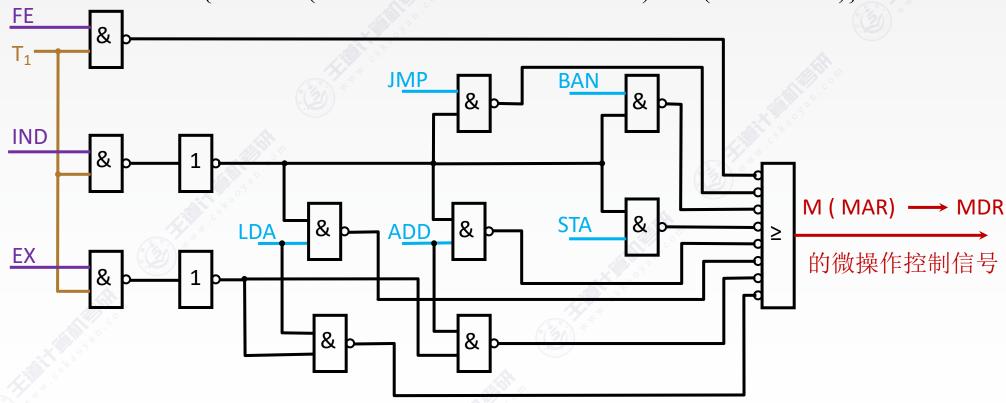
## 微操作信号综合

节拍	状态 条件	微操作命令信号	CLA	СОМ	SHR	CSL	STP	ADD	STA	LDA	JMP	BAN
Т		$PC \longrightarrow MAR$	1	1	1	1	1	1	1	1	1	1
10		$1 \rightarrow R$	1	1	1	1	1	1	1	1	1	1
Т		$M(MAR) \rightarrow MDR$	1	1	1	1	1	1	1	1	1	1
<b>1</b> ]		1	 					20	<u>.</u>	_	<b>.</b>	ا ا ÷
$T_1$		$M(MAR) \rightarrow MDR$						1	1	1	1	1
间址											1	
执行				1 → W					1			
				$M(MAR) \longrightarrow MDR$				1		1		
	T <sub>0</sub>	T <sub>0</sub> 条件	$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	A 条件	$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	A	条件   が採作的マ信号   CEA   COM   SHR   CSL   SHR   T <sub>0</sub>   PC → MAR   1   1   1   1   1   1   1   1   1	条件   根操作的 引	条件   機操作的 受信   CLA   COM   SHK   CSL   SH   ADD   SIA     $T_0$   $PC \rightarrow MAR$   1   1   1   1   1   1   1   1   1	App   条件   一次採作的マーロタ   CLA   COM   SHK   CSL   SH   ADD   SIA   LDA     T <sub>0</sub>	条件   税保行間 マ 信 与   CLA   COM   STR   CSL   STI   ADD   STA   LDA   JMF     $T_0$   PC → MAR   1   1   1   1   1   1   1   1   1

M(MAR)→MDR微操作命令的逻辑表达式:
FE·T₁+IND·T₁(ADD+STA+LDA+JMP+BAN)+EX·T₁(ADD+LDA)
=T₁{FE+IND(ADD+STA+LDA+JMP+BAN)+EX(ADD+LDA)}

### 画出逻辑图

M (MAR) → MDR微操作命令的逻辑表达式: FE·T<sub>1</sub> + IND·T<sub>1</sub>(ADD+STA+LDA+JMP+BAN) + EX·T<sub>1</sub>(ADD+LDA) =T<sub>1</sub>{FE+IND(ADD+STA+LDA+JMP+BAN)+EX(ADD+LDA)}



根据指令操作码、目前的机器周期、节拍信号、机器状态条件,即可确定现在这个节拍下应该发出哪些"微命令"

### 硬布线控制器的设计

#### 设计步骤:

- 1. 分析每个阶段的微操作序列
- 2. 选择CPU的控制方式
- 3. 安排微操作时序
- 4. 电路设计
  - (1) 列出操作时间表
  - (2)写出微操作命令的最简表达式
  - (3)画出逻辑图

#### 硬布线控制器的特点:

指令越多,设计和实现就越复杂,因此一般用于 RISC (精简指令集系统) 如果扩充一条新的指令,则控制器的设计就需要大改,因此扩充指令较困难。 由于使用纯硬件实现控制,因此执行速度很快。微操作控制信号由组合逻辑电路即时产生。



公众号: 王道在线



b站: 王道计算机教育



抖音: 王道计算机考研